(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
06 October 2022 (06.10.2022)

WIPO | PCT

(10) International Publication Number
WO 2022/207961 A1

(51) International Patent Classification:
G06T 17/00 (2006.01) *H04N 21/472* (2011.01)
H04N 21/44 (2011.01) *H04N 21/218* (2011.01)
G06T 19/00 (2011.01) *H04N 21/6587* (2011.01)
H04N 21/854 (2011.01) *G06F 3/01* (2006.01)
G06T 19/20 (2011.01) *G06T 13/40* (2011.01)

(21) International Application Number:
PCT/FI2022/050083

(22) International Filing Date:
11 February 2022 (11.02.2022)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
20215355 29 March 2021 (29.03.2021) FI

(71) Applicant: **NOKIA TECHNOLOGIES OY** [FI/FI]; Karakaari 7, 02610 Espoo (FI).

(72) Inventors: **KONDRAD, Lukasz**; Marienstr. 17, 80331 Munich (DE). **AKSU, Emre, Baris**; Virrenkatu 3 D 20, 33800 Tampere (FI). **ILOLA, Lauri, Aleksi**; Siegestr 29, 80802 Munich (DE). **MALAMAL VADAKITAL, Vinod, Kumar**; Leutokuja 6 A 4, 33870 Tampere (FI).

(74) Agent: **NOKIA TECHNOLOGIES OY** et al.; Ari Aarnio, IPR Department, Karakaari 7, 02610 Espoo (FI).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

WO 2022/207961 A1

(54) Title: A METHOD, AN APPARATUS AND A COMPUTER PROGRAM PRODUCT FOR VIDEO ENCODING AND VIDEO DECODING

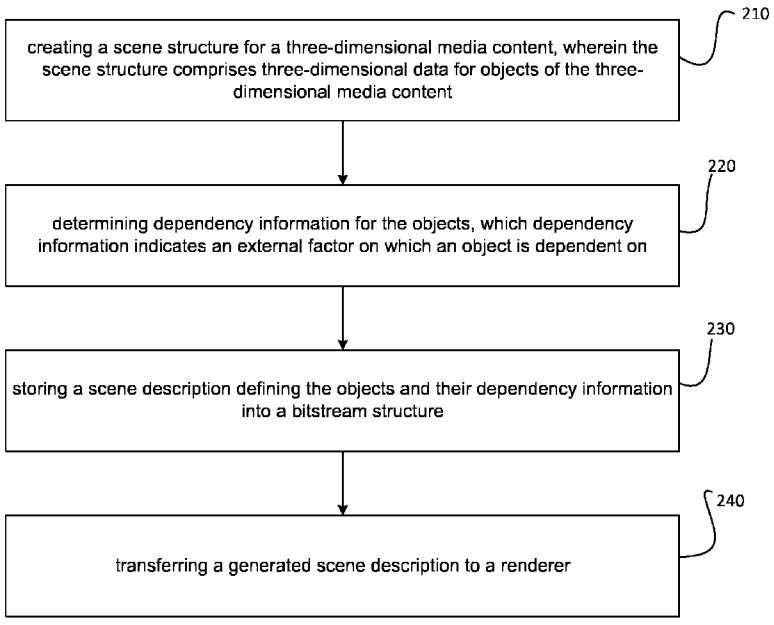


Fig. 2

(57) Abstract: The embodiments relate to a method and a technical equipment for implementing the method. The method comprises (210) creating a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content; (220) determining dependency information for the objects, which dependency information indicates an external factor on which an object is dependent on; (230) storing a scene description defining the objects and their dependency information into a bitstream structure; (240) and transferring a generated scene description to a renderer.

[Continued on next page]

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

A METHOD, AN APPARATUS AND A COMPUTER PROGRAM PRODUCT FOR VIDEO ENCODING AND VIDEO DECODING

Technical Field

The present solution generally relates to coding of immersive media. In particular the present solution relates to scene description for media.

Background

Devices that are able to capture image and video have evolved from devices capturing a limited angular field of view to devices capturing 360-degree content. These devices are able to capture visual and audio content all around them, i.e. they can capture the whole angular field of view, which may be referred to as 360 degrees field of view. More precisely, the devices can capture a spherical field of view (i.e., 360 degrees in all spatial directions). In addition to the new types of image/video capturing devices, also new types of output technologies have been invented and produced, such as head-mounted displays. These devices allow a person to see visual content all around him/her, giving a feeling of being "immersed" into the scene captured by the 360 degrees camera. The new capture and display paradigm, where the field of view is spherical, is commonly referred to as virtual reality (VR) and is believed to be the common way people will experience media content in the future.

Summary

The scope of protection sought for various example embodiments of the invention is set out by the independent claims. The example embodiments and features, if any, described in this specification that do not fall under the scope of the independent claims are to be interpreted as examples useful for understanding various example embodiments of the invention.

Various aspects include a method, an apparatus and a non-transitory computer readable medium comprising a computer program, which are characterized by what is stated in the independent claims. Various details of the example

embodiments are disclosed in the dependent claims and in the corresponding images and description.

According to a first aspect, there is provided a method comprising creating a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content; determining dependency information for the objects, which dependency information indicates an external factor on which an object is dependent on; storing a scene description defining the objects and their dependency information into a bitstream structure; and transferring a generated scene description to a renderer.

According to a second aspect, there is provided an apparatus comprising means for creating a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content; means for determining dependency information for the objects, which dependency information indicates an external factor on which an object is dependent on; means for storing a scene description defining the objects and their dependency information into a bitstream structure; and means for transferring a generated scene description to a renderer.

According to a third aspect, there is provided an apparatus comprising at least one processor, memory including computer program code, the memory and the computer program code configured to, with the at least one processor, cause the apparatus to perform at least the following: creating a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content; determining dependency information for the objects, which dependency information indicates an external factor on which an object is dependent on; storing a scene description defining the objects and their dependency information into a bitstream structure; and transferring a generated scene description to a renderer.

According to a fourth aspect, there is provided a computer program product comprising computer program code configured to, when executed on at least one processor, cause an apparatus or a system to create a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content; determine dependency information for the objects, which dependency information indicates an

external factor on which an object is dependent on; store a scene description defining the objects and their dependency information into a bitstream structure; and transfer a generated scene description to a renderer.

According to an embodiment, the objects are represented as node hierarchy.

According to an embodiment, the dependency information is inherited from parent nodes of the node hierarchy to child nodes of the node hierarchy.

According to an embodiment, the external factor is unavailable during the creation.

According to an embodiment, the external factor comprises one or more of the following: viewer's position, viewer's hand position, viewer's viewport, position of an externally controlled object, position of a real-world object, viewer's rotation, viewer's hand rotation, rotation of an externally controlled object, rotation of a real-world object, viewer's scale, viewer's hand scale, scale of an externally controlled object, scale of a real-world object.

According to an embodiment, the external factor comprises marker information.

According to an embodiment, the bitstream structure is according to Graphics Language Transmission Format (glTF).

According to an embodiment, the dependency information is provided through an extension mechanism of Graphics Language Transmission Format (glTF).

According to an embodiment, the computer program product is embodied on a non-transitory computer readable medium.

Description of the Drawings

In the following, various embodiments will be described in more detail with reference to the appended drawings, in which

Fig. 1 shows an example of a scene structure;

Fig. 2 is a flowchart illustrating a method according to an embodiment; and

Fig. 3 shows an apparatus according to an embodiment.

Description of Example Embodiments

The following description and drawings are illustrative and are not to be construed as unnecessarily limiting. The specific details are provided for a thorough understanding of the disclosure. However, in certain instances, well-known or conventional details are not described in order to avoid obscuring the description. References to one or an embodiment in the present disclosure can be, but not necessarily are, reference to the same embodiment and such references mean at least one of the embodiments.

Reference in this specification to “one embodiment” or “an embodiment” means that a particular feature, structure, or characteristic described in connection with the embodiment in included in at least one embodiment of the disclosure.

Users may consume both videos and images as visual content. However, the consumption of videos and images have been independent on each other. The recent development of applications - such as immersive multimedia - has enabled new use cases where users consume both videos and images together. Immersive multimedia - such as omnidirectional content consumption - is more complex for the end user compared to the consumption of two-dimensional (2D) content. This is due to the higher degree of freedom available to the end user (e.g., three degrees of freedom for yaw, pitch and roll). This freedom also results in more uncertainty.

More and more applications and services are based on three-dimensional (3D) content. In many cases, the 3D content has to be transferred over an internet, and it has to be efficiently rendered on the client side. 3D content that is rendered in client application originates from various sources and is stored in various file formats. In order to render such 3D content, the application has to be capable of reading different input file formats. The scene structure has to be parsed, and the 3D geometry data has to be converted into the format required by the graphics API. This situation has also been identified in the ongoing standardization.

The MPEG standard ISO/IEC 23090 is targeted Coded Representation of Immersive media, where part 14 relates to Scene Description for MPEG Media. As part of ongoing work, the use of scene description formats for enabling immersive augmented reality (AR) and 6-Degrees-of-Freedom (6DoF) experiences are studied. The working group has chosen Khronos ® glTF ™ as the starting point for MPEG scene description initiative for delivery of three-dimensional (3D) and volumetric assets, as well as temporally changing audio-visual media.

The Graphics Language Transmission Format (glTF) is standard file format for 3D scenes and models. It is a JSON (JavaScript Open Notation) based rendering API (Application Programming Interface) agnostic runtime asset delivery format. glTF bridges the gap between three-dimensional (3D) content creation tools and modern 3D applications by providing an efficient, extensible, interoperable format for the transmission and loading of 3D content.

glTF assets are JSON files plus supporting external data. Specifically, a glTF asset is represented by:

- a JSON-formatted file (.gltf) containing a full scene description: node hierarchy, materials, cameras, as well as descriptor information for meshes, animations, and other constructs;
- binary files (.bin) containing geometry and animation data, and other buffer-based data;
- image files (.jpg, .png) for textures.

The JSON formatted file contains information about the binary files that describes how they may be used when uploaded to GPU with minimal processing. This makes the glTF particularly well suitable for runtime delivery, as the assets may be directly copied into GPU memory for the rendering pipeline.

Assets defined in other formats, such as images, may be stored in external files referenced via Uniform Resource Identifier (URI), stored side-by-side in GLB (binary format of graphics language) container, or embedded into the JSON using data URIs.

glTF has been designed to allow extensibility. While the initial base specification supports a rich feature set, there will be many opportunities for growth and

improvement. glTF defines a mechanism that allows the addition of both general-purpose and vendor-specific extension.

The glTF asset contains zero or more scenes, i.e. the set of visual objects to render. Scenes are defined in a scenes array. An additional property, scene (in singular form), identifies which of the scenes in the array is to be displayed at load time.

Figure 1 shows an example of a scene 110 and corresponding glTF nodes in a scene structure 100. Each scene 110 is represented by a list 105 of nodes it contains in scene.nodes.array. Nodes are organized in a parent-child hierarchy known informally as the node hierarchy, as presented by a scene structure 100. A node is called a root node when it does not have a parent. Each root node can also be a separate scene.

Each node can have a transform. Such a transform will define a translation, rotation, and/or scale. This transform is applied to all element attached to the node itself and to all its child nodes. Any node can define a local space transformation. Each node can be targeted for animation (referenced by an animation.channel.target) in the scene.

glTF supports descriptive and skinned animations via key frame animations. Key frame data is stored in buffers and referenced in animations using accessors. glTF also supports animation of morph targets. Animation capabilities in glTF are relatively limited and only animation of node transforms and morph target weights. Animation of texture for example is not currently supported. Key frame timings are defined as seconds that relate to a specific animation timeline. An animation may consist of several key frames, each descripting the state of an object at said keyframes. The animation is created by interpolating node transformations between key frame states.

glTF only defines animation storage, and it does not define any particular runtime behavior, such as: order of playing, auto-start, loops, mapping of timelines, etc. For example, a glTF file may contain two animations, which both contain key frame timing at 1.0. This does not mean that both or either of the key frames would be played at 1.0 seconds in application time. Animations may each contain multiple channels targeting a model's various bones or properties. The client implementation may choose when to play any of the available animations.

All animations are stored in the animations array of the asset. An animation is defined as a set of channels (the channels property) and a set of samplers that specify accessors with key frame data and interpolation method (the samplers property).

glTF 2.0 defines an extension mechanism that allows the base format to be extended with new capabilities. Any glTF object can have an optional extensions property that lists the extensions that are used by that object.

The Potential Improvements to CD of ISO/IEC 23090-14 document defines extensions to glTF 2.0. Among others it defines MPEG media extension, identified by MPEG_media, and MPEG buffer, identified by MPEG_buffer_circular. MPEG media provides an array of MPEG media items used in the scene. MPEG buffer links MPEG media entry that to the buffer element and indicate that the MPEG media is used as the source of data in the buffer. Presence of MPEG buffer extension in glTF buffer element indicates that the buffer could be updated during the duration of the scene rendering and timing information may be provided.

Once the scenes have been created, the scenes described by glTF have predetermined spatial and visual characteristics. Also animations may be stored in glTF, which provide instructions for transforming nodes as designed by the content creator. Temporal application of animations on the nodes, while visually pre-determined, is left to the application or implementation, and may depend on events that happen during runtime, e.g. user clicks a controller button.

Viewer/observer position or motion in the scene does not impact the position of objects nor animations in the scene. A content producer may wish to include objects (nodes) in a scene whose transformation information would be dependent on or relative to one or more external inputs or factors such as (but not limited to):

- viewer position
- viewer's hand position
- viewer's viewport
- position of an externally controlled object, like another person in the scene (for example placing an icon on tope of another person in VR);
- anything else that is related to information received from an external entity.

However, signalling and carriage of properties for such dependency of objects (nodes) on external input for factors is not supported currently by glTF specification nor by any known extensions, including MPEG extensions to glTF. The present embodiments are targeted to such need.

Thus, the present embodiments provide a method for signalling information in the glTF JSON description, which information enables transforming a node based on the external information available during the runtime process, and which information is generally unavailable or not applicable during the content creation process. The solution may be implemented e.g. by a definition of a new extension.

The functionality according to example embodiments enables new use cases and features for 3D scenes stored in glTF or similar descriptions in a file format or data carriage mechanism alike. Such use cases may relate (but are not limited) to the following examples:

Use case 1:

Some people suffer from motion sickness, or simulator sickness, while consuming VR content. The avoid motion sickness, a virtual nose can be rendered on the content, which follows the viewer's viewing direction. The nose may be a node of a scene whose translation, rotation, scale would be dependent of a viewer's position and orientation (i.e. viewer's pose). The dependency of the nose on the viewer's position and orientation may be indicated by the example embodiments.

Use case 2:

While navigating a scene, a placeholder for an MPEG media or realtime communication may be placed in the scene, which then allows following the viewing direction so that it is always visible and perhaps facing the viewer, and at a particular 3D location in space or 2D position on viewport. In a virtual training scenario, such an object could be the volumetric representation of a manufacturing part, which is always visible at e.g. top-right corner of the viewer's viewport, at a fixed distance in 3D space, and locked to the viewer's viewport at fixed 3D position from the viewer. The dependency of the placeholder on the viewport may be indicated by the example embodiments.

The 3D objects as specified using this technique, could take rendering priority and ignore possible depth tests with the content so that the content would always be displayed even when another scene object would be in front of it in 3D space. Alternatively the object could interact with other objects in the scene, when depth collision between the 3D object and the scene occurs due to viewer motion or animations in the scene.

Use case 3:

A content producer could provide additional objects in the scene that are positioned based on external input information, e.g.

- nodes representing hand of a viewer which animation is performed based on external input;
- selection of buttons, where buttons' position is related to current viewer position;
- a video panel or a volumetric 3D object; when externally triggered to be rendered; to be always rendered at a particular location relative to the palm of the hand (e.g. a 2D video panel only playback when the hand's palm is open and looking upward).

Use case 4:

Placing objects related to externally controlled entities. For example, placing an icon on another "real" person in mixed reality.

In each of the above use case examples, a dependency between an additional object on the scene and an external input (also referred to as external information or external factor) can be indicated by means of – for example – an extension of the present embodiments. The external factor may comprise for example one or more of the following 3D transformations: viewer's position, viewer's hand position, viewer's viewport, position of an externally controlled object, position of a real-world object, viewer's rotation, viewer's hand rotation, rotation of an externally controlled object, rotation of a real-world object, viewer's scale, viewer's hand scale, scale of an externally controlled object, scale of a real-world object. It is appreciated that other external factors may be used instead or in addition. Example embodiments on how this may be implemented are discussed in the following. It is to be noticed that the example code in the given format is only an illustrative, and thus should not be unnecessarily interpreted as a limiting example:

According to a first example embodiment, an extension for indicating that the node transformation is dependent on the external information or external factor is defined.

MPEG_node_transformation_external.schema.json can be written as follows:

```
{
        "$schema" : "http://json-schema.org/draft-04/schema",
        "title" : "MPEG_node_transformation_external extension",
        "type" : "object",
        "description": "glTF extension to specify pose in scene is
    dependent on external information",
        "allOf": [ { "$ref": "glTFChildOfRootProperty.schema.json" } ],
        "properties" : {
            "matrix": {
                "uri": {
                    "type": "string",
                    "description": "The uri provides node's source of the
    transformation matrix"
                    "gltf_detailedDescription": "A floating-point 4x4
    transformation matrix stored in column-major order.",
                    "gltf_webgl": "`uniformMatrix4fv()` with the transpose
    parameter equal to false"
                 }
            },
            "rotation": {
                "uri": {
                    "type": "string",
                    "description": "The uri provides node's source of unit
    quaternion rotation in the order (x, y, z, w), where w is the
    scalar.",
                }
                "description": "The node's unit quaternion rotation in the
    order (x, y, z, w), where w is the scalar.",
            },
            "scale": {
                "uri": {
                    "type": "string",
                    "description": "The uri provides node's source of non-
    uniform scale, given as the scaling factors along the x, y, and z
    axes.",
                }
            },
            "translation": {
                "uri": {
                    "type": "string",
                    "description": "The uri provides node's source of
    translation along the x, y, and z axes."
                }
            }
        }
    }
```

An example usage of the extension in an excerpt of glTF JSON is presented below:

```
{
```

```
    "nodes": [
        {
            "name": "Box 1",
            "rotation": [
                0,
                0,
                0,
                1
            ],
            "scale": [
                1,
                1,
                1
            ],
            "translation": [
                -17.7082,
                -11.4156,
                2.0922
            ]
        },
        {
            "name": "Box 2 with extenal transformation",
            "extensions": {
                MPEG_node_transformation_external: {
                    matrix: {
                        uri: "mpeg:transformation:viewer:pose"
                    }
                }
             }
        }

    ]
}
```

A number of URI elements can be defined to indicate which information is used for the transformations. An example is mpeg:transformation:viewer:pose that indicates the viewer pose should be used as input for the transformation of the node.

Possible URI elements may comprise the following (but not limited to them):

- mpeg:transformation:viewer:pose
- mpeg:transformation:viewer:translation
- mpeg:transformation:viewer:rotation
- mpeg:transformation:viewer:scale
- mpeg:transformation:viewer:weights
- mpeg:transformation:viewer:hand:left
- mpeg:transformation:viewer:hand:right
- mpeg:transformation:viewer:location
- mpeg:transformation:viewer:bounding_box

It is to be noticed that the given URI schema is provided as an example and can be any other base URI or schema, as long as the viewer and/or the renderer can uniquely identify and parse the information.

Alternatively, instead of the URI, an enumeration value can be used to provide pre-defined information.

According to a second example embodiment, the extension contains a marker information, which – when detected, by an application – would indicate the position of the node. In other words, the node transformation is fixed to an external location that corresponds to the marker. An example of such marker is a QR code.

```
{
        "$schema" : "http://json-schema.org/draft-04/schema",
        "title" : "MPEG_node_transformation_external extension",
        "type" : "object",
        "description": "glTF extension to specify position of scene is
    dependent on external information",
        "allOf": [ { "$ref": "glTFChildOfRootProperty.schema.json" } ],
        "properties" : {
            "matrix": {
                "marker_type": {
                    "type": "string",
                    "description": "The type of marker that will corespond
    to the position oft he node in the scene."
                }
                "marker_data_uri": {
                    "type": "string",
                    "description": "The uri that provides the description
    oft he marker"
                }

                "description": "The renderer will expect an matrix that is
    calculated based on the viewer position and external marker location"
                "gltf_detailedDescription": "A floating-point 4x4
    transformation matrix stored in column-major order.",
                "gltf_webgl": "`uniformMatrix4fv()` with the transpose
    parameter equal to false"
                 }
            }
        }
    "required": [ "marker_data_uri"]
    }
```

An example usage of the extension in an excerpt of glTF JSON is presented below:

```
{
    "nodes": [
        {
            "name": "Box 1",
            "rotation": [
                0,
```

```
                    0,
                    0,
                    1
                ],
                "scale": [
                    1,
                    1,
                    1
                ],
                "translation": [
                    -17.7082,
                    -11.4156,
                    2.0922
                ]
            },
            {
                "name": "Box 2 with extenal transformation",
                "extensions": {
                    MPEG_node_transformation_external: {
                        matrix: {
                            marker_type: "mpeg:sd:geolocation"
                            marker_data_uri: "data:text/plain;charset=UTF-8;
35.1592;-98.4422;410"
                        }
                    }
                }
            }

        ]
}
```

The renderer will expect a matrix that is calculated based on the viewer position and external marker location. Marker information are provided by the extension.

marker_type provides type of the marker e.g. geographic location, QR code data or any other 2D encoded visual code or detectable marker (e.g. a particular shaped light source).

marker_data_uri may be:

- a URI to an external file in format that is compliant with marker type, or
- a URI with embedded base64-encoded data in the following format data: [<mediatype>][;base64],<data>
    - An example of plain location encoded in a text is "*data:text/plain;charset=UTF-8;35.1592;-98.4422;410*" the format is (decimal latitude);(decimal longitude);([optional]elevation in meters)

The type may be not present and just by URI the application may determine the location of an object.

- Using RFC 5870, an example of such uri would be geo:37.786971,-122.399677;u=35

In another example, such marker data may be stored as URI items or deductive information items in an ISOBMFF file. Such items may then be referenced using ISOBMFF or HEIF MIME URL referencing mechanisms.

According to a third example embodiment, a URI information described in the first and the second example embodiment could be listed by existing MPEG_media extension and reference by source element of MPEG_buffer_circular extension. The buffer element with the MPEG_buffer_circular extension would be reference by bufferView of animation output sampler, which provide the transformation information, e.g. translation in the example below. In this scenario, the input element of the animation sampler would be set to predefined value (e.g. minus 1), or will be absent, indicating that timing information is provided outside the glTF. As an example, timing information can be provided as an update timing of the output sampler buffer.

An example usage of the idea in an excerpt of glTF JSON is presented below:

```
{
    "animations": [
        {
            "name": "Animate a node 1 rotation channel based on the
extenal information",
            "channels": [
                {
                    "sampler": 0,
                    "target": {
                        "node": 1,
                        "path": "translation"
                    }
                },
            ],
            "samplers": [
                {
                    "interpolation": "LINEAR",
                    "output": 0
                }
            ]
        }
    ],
    "accessors": [
        {
            "bufferView": 0,
            "byteOffset": 0,
            "componentType": 5126,
            "type": "VEC3"
        }
    ],
    "buffers": [
```

```
        {
             "extensions": {
                 "MPEG_circular_buffer": {
                 "source": 0
        }
      }        }
    ],
    "bufferViews": [
        {
            "buffer": 0,
            "byteLength": 96,
            "byteOffset": 0
        }
    ],
     "extensions": {
        "MPEG_media": {
          "media": [{
              "name": "dynamic_scene_data1",
              "alternatives": [{
                  "uri": "mpeg:transformation:viewer:rotation"
               }]
          }]
        }
    }
}
```

According to a fourth example embodiment, for viewer dependent content, alpha coded masks may be provided during composition to generate viewer dependent objects.

One or two masks (for left and right eye) may be provided.

During rendering, the viewport is rendered as usual, and at the final stage of rendering an additional compositing is applied, where the masks are used to composite the viewer dependent objects into the rendered image.

Depending on the importance of how photo-realistic the final rendering is required to be, e.g. on the virtual nose, additional information like specular and normal maps for the virtual nose may also be provided.

According to a fifth example embodiment, for all the above – from first to fourth – example embodiments, a streaming media manifest technology such as DASH may be utilized to signal the presence of such nodes in the scene description. Therefore, the renderers being capable of such viewer dependent rendering features can select them, while the others can select alternative representations. This may be achieved by signalling an EssentialProperty or non-Essential property in the DASH manifest,

and related value in the DASH representation level. The value of this EssentialProperty can be a Boolean which indicates that the representation utilizes viewer dependent 3D objects.

In an example, the file which stores the glTF information may have additional brand which indicates the presence of viewer dependent 3D rendering data structures.

In above examples for indicating node dependency on external information has be discussed.

The method according to an embodiment is shown in Figure 2. The method generally comprises creating 210 a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content; determining 220 dependency information for the objects, which dependency information indicates an external factor on which an object is dependent on; storing 230 a scene description defining the objects and their dependency information into a bitstream structure; and transferring 240 a generated scene description to a renderer. Each of the steps can be implemented by a respective module of a computer system.

An apparatus according to an embodiment comprises means for creating a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content; means for determining dependency information for the objects, which dependency information indicates an external factor on which an object is dependent on; means for storing a scene description defining the objects and their dependency information into a bitstream structure; and means for transferring a generated scene description to a renderer. The means comprises at least one processor, and a memory including a computer program code, wherein the processor may further comprise processor circuitry. The memory and the computer program code are configured to, with the at least one processor, cause the apparatus to perform a method according to various embodiments.

An apparatus according to an embodiment is illustrated in Figure 3. The apparatus is a user equipment (also referred to as “client”) for the purposes of the present embodiments, i.e. for content authoring. The apparatus 90 comprises a main processing unit 91, a memory 92, a user interface 94, a communication

interface 93. The apparatus according to an embodiment, shown in Figure 3, also comprises a camera module 95. Alternatively, the apparatus may be configured to receive image and/or video data from an external camera device over a communication network. The memory 92 stores data including computer program code in the apparatus 90. The computer program code is configured to implement the method according various embodiments by means of various computer modules. The camera module 95 or the communication interface 93 receives data, in the form of images or video stream, to be processed by the processor 91. The communication interface 93 forwards processed data, i.e. the image file, for example to a display of another device, such a virtual reality headset. When the apparatus 90 is a video source comprising the camera module 95, user inputs may be received from the user interface.

An example of a device for content consumption, i.e. an apparatus according to another embodiment, is a virtual reality headset, such as a head-mounted display (HMD) for stereo viewing. The head-mounted display may comprise two screen sections or two screens for displaying the left and right eye images. The displays are close to the eyes, and therefore lenses are used to make the images easily viewable and for spreading the images to cover as much as possible of the eyes' field of view. The device is attached to the head of the user so that it stays in place even when the user turns his head. The device may have an orientation detecting module for determining the head
movements and direction of the head. The head-mounted display is able to show omnidirectional content (3DOF content) of the recorded/streamed image file to a user.

The various embodiments can be implemented with the help of computer program code that resides in a memory and causes the relevant apparatuses to carry out the method. For example, a device may comprise circuitry and electronics for handling, receiving and transmitting data, computer program code in a memory, and a processor that, when running the computer program code, causes the device to carry out the features of an embodiment. Yet further, a network device like a server may comprise circuitry and electronics for handling, receiving and transmitting data, computer program code in a memory, and a processor that, when running the computer program code, causes the network device to carry out the features of an embodiment.

If desired, the different functions discussed herein may be performed in a different order and/or concurrently with other. Furthermore, if desired, one or more of the above-described functions and embodiments may be optional or may be combined.

Although various aspects of the embodiments are set out in the independent claims, other aspects comprise other combinations of features from the described embodiments and/or the dependent claims with the features of the independent claims, and not solely the combinations explicitly set out in the claims.

It is also noted herein that while the above describes example embodiments, these descriptions should not be viewed in a limiting sense. Rather, there are several variations and modifications, which may be made without departing from the scope of the present disclosure as, defined in the appended claims.

Claims:

1. A method, comprising:
- creating a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content;
- determining dependency information for the objects, which dependency information indicates an external factor on which an object is dependent on;
- storing a scene description defining the objects and their dependency information into a bitstream structure; and
- transferring a generated scene description to a renderer.

2. The method according to claim 1, wherein the objects are represented as node hierarchy.

3. The method according to claim 1 or 2, wherein the external factor is unavailable during the creation.

4. The method according to claim 3, wherein the external factor comprises one or more of the following: viewer's position, viewer's hand position, viewer's viewport, position of an externally controlled object, position of a real-world object, viewer's rotation, viewer's hand rotation, rotation of an externally controlled object, rotation of a real-world object, viewer's scale, viewer's hand scale, scale of an externally controlled object, scale of a real-world object.

5. The method according to claim 3, wherein the external factor comprises marker information.

6. The method according to any of the claims 1 to 5, wherein the bitstream structure is according to Graphics Language Transmission Format (glTF).

7. An apparatus comprising:
- means for creating a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content;
- means for determining dependency information for the objects, which dependency information indicates an external factor on which an object is dependent on;

- means for storing a scene description defining the objects and their dependency information into a bitstream structure; and
- means for transferring a generated scene description to a renderer.

8. The apparatus according to claim 7, wherein the objects are represented as node hierarchy.

9. The apparatus according to claim 8, wherein the dependency information is inherited from parent nodes of the node hierarchy to child nodes of the node hierarchy.

10. The apparatus according to claim 7 or 8 or 9, wherein the external factor is unavailable during the creation.

11. The apparatus according to claim 10, wherein the external factor comprises one or more of the following: viewer's position, viewer's hand position, viewer's viewport, position of an externally controlled object, position of a real-world object, viewer's rotation, viewer's hand rotation, rotation of an externally controlled object, rotation of a real-world object, viewer's scale, viewer's hand scale, scale of an externally controlled object, scale of a real-world object.

12. The apparatus according to claim 10, wherein the external factor comprises marker information.

13. The apparatus according to any of the claims 7 to 12, wherein the bitstream structure is according to Graphics Language Transmission Format (glTF).

14. The apparatus according to any of the claim 13, wherein the dependency information is provided through an extension mechanism of Graphics Language Transmission Format (glTF).

15. An apparatus comprising at least one processor, memory including computer program code, the memory and the computer program code configured to, with the at least one processor, cause the apparatus to perform at least the following:
- create a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content;

- determine dependency information for the objects, which dependency information indicates an external factor on which an object is dependent on;
- store a scene description defining the objects and their dependency information into a bitstream structure; and
- transfer a generated scene description to a renderer.

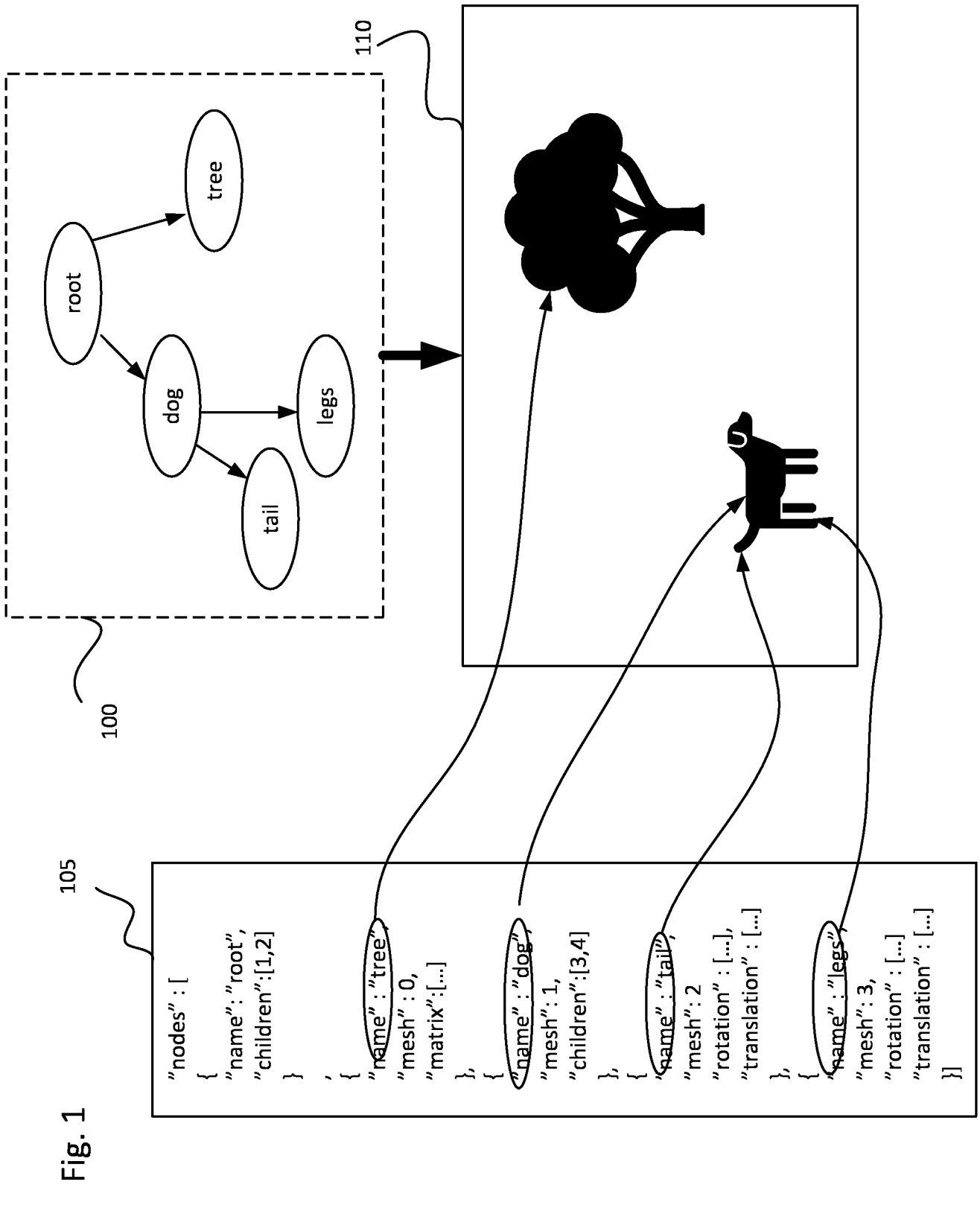

root
tree
dog
legs
tail
100
110
105
"nodes" : [
{
"name": "root",
"children":[1,2]
},
{
"name" : "tree",
"mesh" : 0,
"matrix":[...]
},
{
"name" : "dog",
"mesh": 1,
"children":[3,4]
},
{
"name" : "tail",
"mesh": 2
"rotation" : [...],
"translation" : [...]
},
{
"name" : "legs",
"mesh": 3,
"rotation" : [...]
"translation" : [...]
}]

Fig. 1

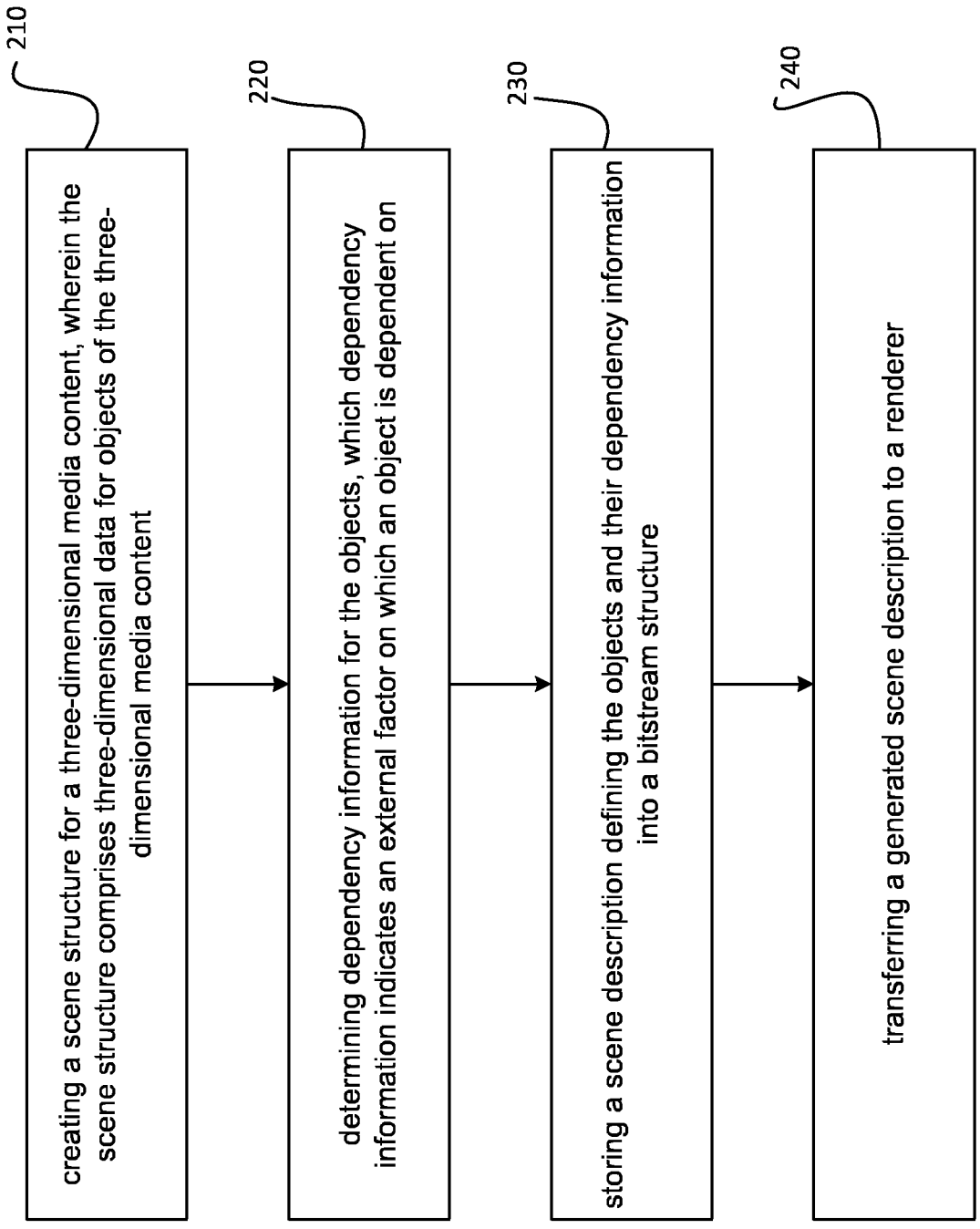

210
creating a scene structure for a three-dimensional media content, wherein the scene structure comprises three-dimensional data for objects of the three-dimensional media content
220
determining dependency information for the objects, which dependency information indicates an external factor on which an object is dependent on
230
storing a scene description defining the objects and their dependency information into a bitstream structure
240
transferring a generated scene description to a renderer

Fig. 2

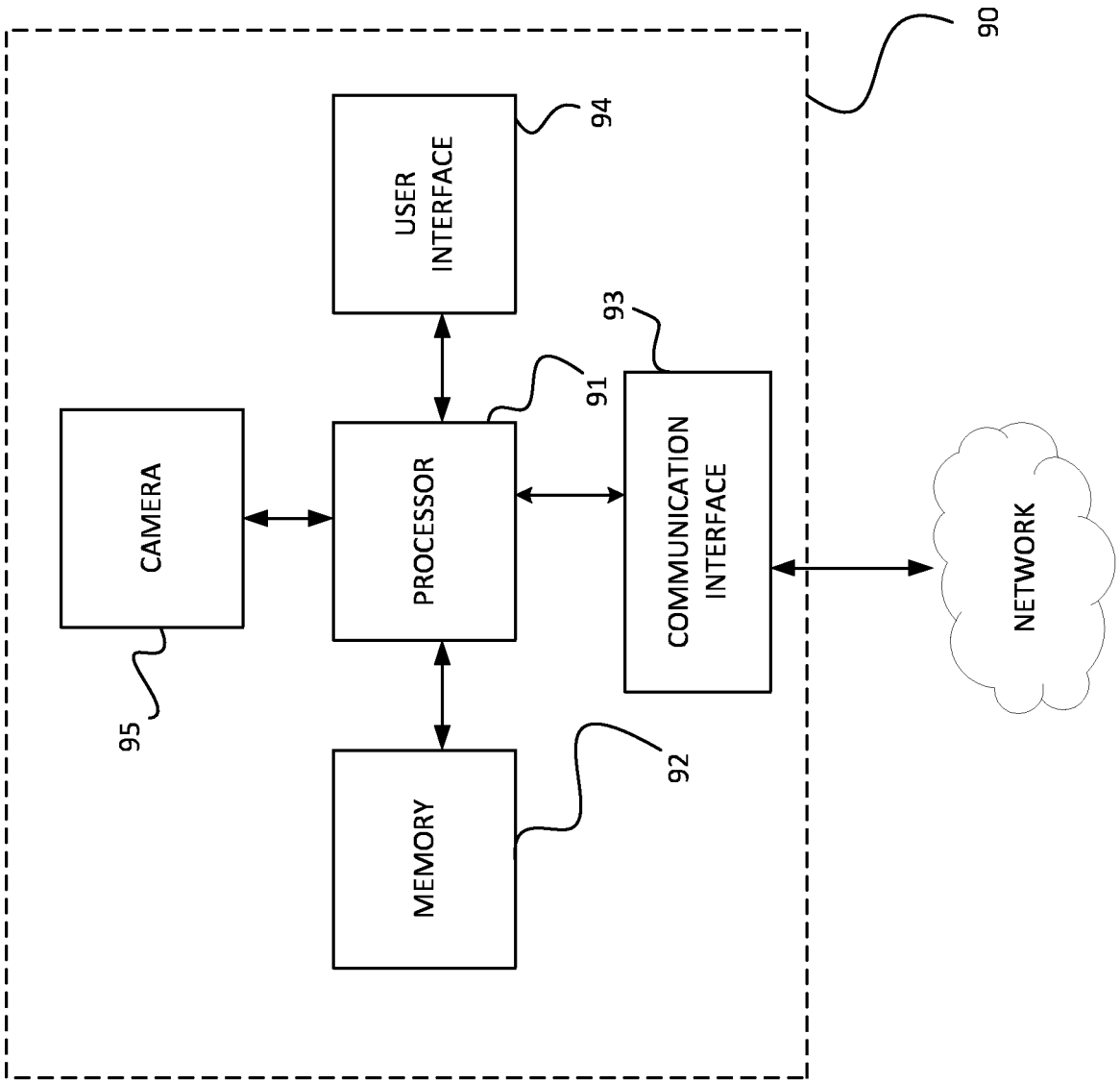

90
USER INTERFACE
94
93
91
CAMERA
PROCESSOR
COMMUNICATION INTERFACE
NETWORK
95
MEMORY
92

Fig. 3

INTERNATIONAL SEARCH REPORT

International application No.
PCT/FI2022/050083

A. **CLASSIFICATION OF SUBJECT MATTER**

See extra sheet

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC: G06T, G06F, H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

FI, SE, NO, DK

Electronic data base consulted during the international search (name of data base, and, where practicable, search terms used)

EPODOC, EPO-Internal full-text databases, Full-text translation databases from Asian languages, WPIAP, COMPDX, INSPEC, TDB, NPL, Internet, IPRally

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	'Potential improvement on ISO/IEC 23090-14 Scene Description for MPEG Media', MPEG document management system [online], 133rd meeting, document w19952, 2021-01-25, [retrieved on 2022-05-13]. Retrieved from <https://dms.mpeg.expert/> sections 1, 4.1, 4.2, 4.3.1, 5.2.1.2, 6.1, 8.3.1; Figs. 1-2, 4	1-15
X	US 2021090315 A1 (GLADKOV DENIS [US] et al.) 25 March 2021 (25.03.2021) paragraphs [0005], [0032]-[0033], [0039]-[0040], [0042], [0051], [0073], [0080], [0092], [0096]; Figs. 1A-1B, 5	1-15
X	US 2012105473 A1 (BAR-ZEEV AVI [US] et al.) 03 May 2012 (03.05.2012) paragraphs [0003], [0066]-[0067], [0082], [0086], [0112], [0115]-[0116], [0139], [0141]; Figs. 9, 12, 16	1, 4-5, 7, 11-12, 15
A	US 5764241 A (ELLIOTT CONAL M [US] et al.) 09 June 1998 (09.06.1998) abstract; column 8, lines 6-23	1-15

☐ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"D" document cited by the applicant in the international application

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
13 May 2022 (13.05.2022)	23 May 2022 (23.05.2022)

Name and mailing address of the ISA/FI
Finnish Patent and Registration Office
FI-00091 PRH, FINLAND
Facsimile No. +358 29 509 5328

Authorized officer
Mikko Flykt
Telephone No. +358 29 509 5000

US 2021090315 A1	25/03/2021	US 11132827 B2	28/09/2021
		US 2022005254 A1	06/01/2022
US 2012105473 A1	03/05/2012	US 9348141 B2	24/05/2016
		CN 102591449 A	18/07/2012
		CN 102591449 B	21/09/2016
		HK 1171532 A1	28/03/2013
		US 2016267717 A1	15/09/2016
		US 9710973 B2	18/07/2017
US 5764241 A	09/06/1998	WO 9720267 A1	05/06/1997

INTERNATIONAL SEARCH REPORT

International application No.
PCT/FI2022/050083

CLASSIFICATION OF SUBJECT MATTER

IPC
G06T 17/00 (2006.01)
H04N 21/44 (2011.01)
G06T 19/00 (2011.01)
H04N 21/854 (2011.01)
G06T 19/20 (2011.01)
H04N 21/472 (2011.01)
H04N 21/218 (2011.01)
H04N 21/6587 (2011.01)
G06F 3/01 (2006.01)
G06T 13/40 (2011.01)